IOS Press

Towards a Framework for Approximate Ontologies

Patrick Doherty*

Michał Grabowski†

Witold Łukaszewicz‡

Andrzej Szałas‡

Abstract. Currently, there is a great deal of interest in developing tools for the generation and use of ontologies on the WWW. These knowledge structures are considered essential to the success of the semantic web, the next phase in the evolution of the WWW. Much recent work with ontologies assumes that the concepts used as building blocks are *crisp* as opposed to approximate. It is a premise of this paper that approximate concepts and ontologies will become increasingly more important as the semantic web becomes a reality. We propose a framework for specifying, generating and using approximate ontologies. More specifically, (1) a formal framework for defining approximate concepts, ontologies and operations on approximate concepts and ontologies is presented. The framework is based on intuitions from rough set theory; (2)algorithms for automatically generating approximate ontologies from traditional crisp ontologies or from large data sets together with additional knowledge is presented. The knowledge will generally be related to similarity measurements between individual objects in the data sets, or constraints of a logical nature which rule out particular constellations of concepts and dependencies in generated ontologies.

The techniques for generating approximate ontologies are parameterizable. The paper provides specific instantiations and examples.

Keywords: approximate concept, approximate reasoning, approximation space, category, concept, approximate concept, approximate ontology, ontology, tolerance space, tolerance-based ontology

*Department of Computer and Information Science, Linköping University, S-581 83 Linköping, Sweden, e-mail: patdo@ida.liu.se. Supported by the Wallenberg Foundation.

†The College of Economics and Computer Science, Olsztyn, Poland e-mail: mich@mimuw.edu.pl. Supported by the Wallenberg Foundation.

‡The College of Economics and Computer Science, Olsztyn, Poland, and Department of Computer and Information Science, Linköping University, S-581 83 Linköping, Sweden, e-mail: witlu@ida.liu.se, andsz@ida.liu.se. Supported by the Wallenberg Foundation and KBN grant 8 T11C 00919.

1. Introduction

With the inception of the World-Wide Web (WWW), a distributed information infrastructure has been set up containing a vast repository of information resources. This infrastructure is designed primarily for human use, with little support for the deployment of software agents which can take advantage of these information resources to assist humans in accomplishing various information processing and gathering tasks. The next stage in the evolution of the the WWW is to enhance the current infrastructure with support for explicit, machine accessible descriptions of information content on the Web. These machine accessible descriptions of information content should be usable and understandable by machines, in particular software agents. Tim Berners-Lee has used the term *Semantic Web* – a web of data that can be processed directly or indirectly by machines[1], to describe this next phase in the evolution of the Web.

The meaning or semantics of diverse information content has to be accessible to software agents for use and reasoning if sophisticated knowledge intensive tasks are to be automated in the Web context. Most importantly, just as humans cooperate and communicate in a common language and conceptual space in order to achieve complex tasks, so will software agents, both with other software agents and with humans. There is a great deal of research activity in this area, particularly in providing the necessary tools to support communication among agents and construction and use of shared *ontologies*.

Webster's dictionary defines *ontology* as,

> "the branch of metaphysics dealing with the nature of being or reality".

In artificial intelligence, more specifically the sub-discipline of knowledge representation, the term is used somewhat more pragmatically to describe how we choose to "slice up" reality and represent these choices in representational structures used for reasoning about agent environments at various levels of abstraction. One common way of "slicing" or conceptualizing is to specify a base set of individuals, properties, relations and dependencies between them. This choice is particularly amenable to the use of logic as a representational tool.

In knowledge engineering circles, the term "ontology" has taken on a related but somewhat different meaning as explicit computational structures which "provide a machine-processible semantics of information sources that can be communicated between different agents (software and human)"[5]. Ontologies are used to facilitate the construction of domain models and to provide the vocabulary and relations used in this process. Gruber [6] provides the following definition:

> "An ontology is a formal, explicit specification of a shared conceptualization."

The intention is that ontologies should facilitate the use of knowledge sharing and reuse among software agents and humans alike.

Just as the Web is currently a heterogeneous collection of information sources, it is reasonable to assume that the future semantic web will include a collection of heterogeneous domain-specific ontologies, sometimes with semantic or syntactic overlap and sometimes not. It is likely that many concepts used in ontologies, be they local or global, will be difficult to define other than approximately. In fact, the process of defining ontologies will often need to be automated and tools will be required to automatically generate ontologies based, for instance, on an initial universe of objects or individuals and some knowledge about them. One example of knowledge might be information about the *similarity* between individuals. From this one would try to induce natural categories through various techniques, including those proposed in this paper.

Even though there are differing views as to what ontologies are, or how they may be appropriately represented, there are some common conceptual denominators. What is common to the approaches is that they are based on "concepts" (sometimes called "categories") and the relations "to be more general" or "to be more specific" between concepts. Those relations form a hierarchy of concepts (ontology) in a form of a tree, lattice or an acyclic graph.

Concepts are usually assumed to be precise (crisp), but as stated previously, we believe that approximate concepts and ontologies will be necessary. In fact, approximate ontologies provide an ideal basis for dealing with some very difficult, but necessary operations on ontologies such as merging, comparing, or incrementally modifying ontologies.

The purpose of this paper is to propose a framework for defining and automatically generating approximate ontologies from traditional crisp ontologies or from large amounts of data with knowledge about similarities between individual data points. More specifically,

- a formal framework for defining approximate concepts, ontologies and operations on approximate concepts and ontologies will be proposed. The framework is based on intuitions from rough set theory;
- algorithms for automatically generating approximate ontologies from large data sets together with additional knowledge will be proposed. The knowledge will generally be related to similarity measurements between individual objects in the data sets, or constraints of a logical nature which rule out particular constellations of concepts and dependencies in generated ontologies.

Before describing the structure of the paper, some intuitions about rough sets would be useful. Rough sets were first introduced by Pawlak [14]. The basic idea centers around the notion of indiscernability among individuals in a domain of discourse. Suppose a particular domain of discourse is given in addition to some knowledge which allows one to partially discern different individuals from each other based on that knowledge. The domain of discourse can then be partitioned into equivalence classes, where individuals in the same equivalence class are indiscernible from each other for all practical purposes due to the limited knowledge provided to discern among individuals. Now, suppose one is given a set X in the domain of discourse. From the partition on individuals based on indiscernability, one can define a lower and upper approximation of X in terms of the equivalence classes in the partition. The lower approximation consists of the union of those equivalence classes which are a subset of X. The upper approximation is the union of those equivalence classes which are not disjoint with X. In this manner, rough sets can be defined in terms of their upper and lower approximations. This is the kernel idea from which we will proceed.

In section 2, *approximation spaces* are introduced. These provide a generalization of the idea of rough set by permitting more general types of grouping or partitioning of the domain of discourse, not necessarily in terms of indiscernability. Using approximation spaces, upper and lower approximations on sets in the domain of discourse are defined. These sets are called *approximate sets*. An *approximate concept* is simply an approximate set. In section 3, *approximate ontologies* are defined along with the definition of a *most specific approximate concept* and a *most general approximate concept* relative to a specific approximate concept. A refinement of an approximate ontology to one that is *strict* is then introduced. Given a strict ontology and an individual, it is then possible to uniquely identify the most specific approximate concept to which the individual belongs and an algorithm is presented which does this. These sections provide a basis for a formal framework for approximate ontologies.

Section 5 introduces two means of generating approximate ontologies from existing data. In the first, an approximate ontology can be generated from a crisp set-based ontology together with an approximation space. This is interesting because it allows one to approximate already existing ontologies based on additional information about similarity or discernibility among individuals. In the second approach, no additional structure is assumed on the original domain of discourse other than that of an approximation space. From a raw data set and approximation space, one can generate an approximate ontology.

Section 7 considers a specific type of approximation space called a *tolerance space* which represents similarity criteria for generating neighborhoods among individuals in a domain of discourse. Special types of approximate ontologies called *tolerance-based ontologies* are defined and algorithms to generate them from information systems are presented. This is a particularly interesting approach since it creates a bridge to research on categorization and techniques for generating categories found in the machine learning and cognitive psychology literature where similarity among individuals and prototypes is essential. For example, in the machine learning literature, concepts are often discovered using a technique called *clustering*. Clusters are sets of objects close to each other wrt some distance measure, assuming there is no continuity between objects in clusters and objects outside of clusters. This idea has been discussed in the context of description logics in [9], however in a much more restricted context than that considered in our paper.

2. Approximation Spaces and Approximate Sets

Throughout the paper we use 2^U to denote the set of all subsets of a domain U, and $\Re$ to denote the set of reals.

Approximation spaces are frequently considered in the literature (see, e.g., [2, 4, 12, 16, 17]).

Definition 2.1. By an *approximation space* we understand a tuple $AS = \langle U, I, \nu \rangle$, where:

- U is a nonempty set of objects, called a *domain*
- $I : U \longrightarrow 2^U$ is an *uncertainty function*
- $\nu : 2^U \times 2^U \longrightarrow \Re$ is an inclusion function with $\Re$ restricted to the interval $[0, 1]$. ■

The intuitive meaning of $I(u)$ is the set of objects "similar" to u, in some sense. The inclusion function $\nu(U_1, U_2)$ provides a degree of inclusion of U_1 in U_2.

Approximation spaces are used as a basis for defining the lower and upper approximations of a set (see, e.g., [2, 17]).

Definition 2.2. Let $AS = \langle U, I, \nu \rangle$ be an approximation space and let $S \subseteq U$. The *lower and upper approximation of S wrt AS*, denoted respectively by S_{AS^+} and $S_{AS^\oplus}$, are defined by

$$S_{AS^+} = \{u \in U : \nu(I(u), S) = 1\}$$
$$S_{AS^\oplus} = \{u \in U : \nu(I(u), S) > 0\}.$$ ■

The intuitive meaning of S_{AS^+} and $S_{AS^\oplus}$ is the following, where uncertainty and inclusion of an approximation space AS is taken into account:

- the lower approximation S_{AS^+} of S consists of elements that are surely in S
- the upper approximation $S_{AS^\oplus}$ of S consists of elements that might be in S.

In consequence, the complement of the upper approximation of a set, $-S_{AS^\oplus}$, consists of elements that are surely not in S.

Definition 2.3. Let $AS = \langle U, I, \nu \rangle$ be an approximation space. By an *approximate set* over AS we mean any pair of the form $\langle S_{AS^+}, S_{AS^\oplus} \rangle$, where $S \subseteq U$. By a *crisp* set (or a *set*, for brevity[1]) we mean any approximate set $\langle S_{AS^+}, S_{AS^\oplus} \rangle$ such that $S_{AS^+} = S_{AS^\oplus}$. ▪

Definition 2.4. Let $AS = \langle U, I, \nu \rangle$ be an approximation space. Let $S = \langle S_1, S_2 \rangle$ and $T = \langle T_1, T_2 \rangle$ be approximate sets over AS. We say that S is *included* in T, denoted by $S \sqsubseteq T$, if $\nu(S_1, T_1) = 1$ and $\nu(S_2, T_2) = 1$. ▪

For the sake of simplicity, we often restrict ourselves to standard approximation spaces defined as follows.

Definition 2.5. We shall say that the approximation space $AS = \langle U, I, \nu \rangle$ is *standard* iff its domain U is finite and its inclusion function is the *standard inclusion function*, defined for any $S, T \subseteq U$ as follows:

$$\nu(S,T) \stackrel{\text{def}}{=} \begin{cases} \dfrac{|\, S \cap T \,|}{|\, S \,|} & \text{when } S \neq \emptyset \\ 1 & \text{when } S = \emptyset. \end{cases}$$ ▪

Since the inclusion function ν is always known in the case of standard approximation spaces, such spaces are denoted by $\langle U, I \rangle$ rather than by $\langle U, I, \nu \rangle$. Observe that for any standard approximation space $\langle U, I \rangle$ and for any $S, T \subseteq U$,

$$\nu(S,T) = 1 \text{ iff } S \subseteq T$$

Thus, in the case of standard approximation spaces, we have the following alternative characterization of $\sqsubseteq$ and approximations.

Proposition 2.1. If $AS = \langle U, I \rangle$ is a standard approximation space, then

1. $\langle S_1, S_2 \rangle \sqsubseteq \langle T_1, T_2 \rangle$ iff $S_1 \subseteq T_1$ and $S_2 \subseteq T_2$
2. $S_{AS^+} = \{u \in U : I(u) \subseteq S\}$ and $S_{AS^\oplus} = \{u \in U : I(u) \cap S \neq \emptyset\}$. ▪

Unless stated otherwise, we will use standard approximation spaces throughout the paper.

[1] It is easily observed that the notion of a crisp set coincides with the notion of a set in the classical sense.

3. Concepts and Approximate Concepts

Let us first discuss the representation of concepts. In the machine learning literature, a concept is often represented as a set of examples. Examples are marked positive, negative or unknown, meaning that they are, are not, or it is unknown whether, they belong to a particular concept, respectively.

Throughout the paper we concentrate on concept representations defined only using examples marked positively and we generate ontologies based on positive knowledge. The techniques we propose generate additional concepts given initial sets of positive examples and additional knowledge about these examples. For example, if the set of examples consists of animals, we might be interested in discovering concepts such as “dogs”, ”cats”, “birds” “mammals”, etc., together with relationships between them, which include ”to be more specific than” and “to be more general than”. Some of the newly generated concepts will be disjoint, while some are generalizations or specializations of other concepts. The same methodology can be applied to negative knowledge, but this addition will not be considered in this paper.

Typically, sets of examples are not provided explicitly, but rather through the use of representations from which membership functions can be efficiently computed. In what follows we will only assume membership functions for particular sets of objects that can be generated using tractable algorithms.

Definition 3.1. Let $AS = \langle U, I, \nu\rangle$ be an approximation space. By an *approximate concept* over AS we understand any approximate set $\langle S_{AS^+}, S_{AS^\oplus}\rangle$ over AS.[2] By a (*crisp*) *concept* we understand any crisp set over AS. By *bottom* and *top* concepts over U, denoted by BOTTOM and TOP, we understand the (crisp) concepts $\langle\emptyset, \emptyset\rangle$ and $\langle U_{AS^+}, U\rangle$, respectively.[3]

If C, D are approximate concepts such that $C \sqsubseteq D$ then we say that C is *more specific* than D (is *a specialization* of D) and that D is *more general* than C (is a *generalization* of C). ■

Example 3.1. Consider the standard approximation space $AS = \langle\{or, r, dr, y, g, gr\}, I\rangle$, where:

- elements of the domain stand for colors “orange red”, “red”, “dark red”, “yellow”, “gold” and “goldenrod”, respectively
- $I(or) = \{or, r\}$, $I(r) = \{or, r, dr\}$, $I(dr) = \{r, dr\}$, $I(y) = \{y, g\}$, $I(g) = I(gr) = \{g, gr\}$.

Sets $\{or, r, dr\}$, $\{g, gr\}$ and $\{y, g, gr\}$ are crisp concepts, since:

$$\begin{array}{lclcl}\{or, r, dr\} & = & \{or, r, dr\}_{AS^+} & = & \{or, r, dr\}_{AS^\oplus},\\ \{g, gr\} & = & \{g, gr\}_{AS^+} & = & \{g, gr\}_{AS^\oplus}\\ \{y, g, gr\} & = & \{y, g, gr\}_{AS^+} & = & \{y, g, gr\}_{AS^\oplus}.\end{array}$$

The pair $\langle\{dr\}, \{or, r, dr\}\rangle$ is an approximate concept, since

$$\{dr\} = \{r, dr\}_{AS^+} \text{ and } \{or, r, dr\} = \{r, dr\}_{AS^\oplus}.$$

The pair $\langle\{dr\}, \{or, dr\}\rangle$ is not an approximate concept since, in this case, $\{or, dr\}$ is not an upper approximation of any set. ■

[2] It is assumed of course, that there is an $S \subseteq U$ where S_{AS^+} is the lower approximation of S and $S_{AS^\oplus}$ is the upper approximation of S. See the example 3.1 below.

[3] Note that $\langle\emptyset, \emptyset\rangle$ and $\langle U_{AS^+}, U\rangle$ are approximate concepts, since $\langle\emptyset, \emptyset\rangle = \langle\emptyset_{AS^+}, \emptyset_{AS^\oplus}\rangle$ and $\langle U_{AS^+}, U\rangle = \langle U_{AS^+}, U_{AS^\oplus}\rangle$.

Remark 3.1. Observe that crisp concepts can be identified with crisp sets. In the rest of the paper, crisp concepts are then denoted as single sets rather than pairs of sets. ▪

4. Approximate Ontologies

Let us start with a definition for an approximate ontology.

Definition 4.1. Let $AS = \langle U, I, \nu \rangle$ be an approximation space. By an *approximate ontology* over AS we understand the tuple $\mathcal{O}_{AS} = \langle U, \mathcal{C}, \sqsubseteq \rangle$, where $\mathcal{C}$ is a set of approximate concepts over AS such that (at least) $\textsc{Bottom}, \textsc{Top} \in \mathcal{C}$. By a *crisp ontology* over U we mean any approximate ontology consisting of crisp concepts only.

If AS is a standard approximation space then an approximate ontology over AS is called the *standard approximate ontology*. ▪

Example 4.1. In the examples below we assume an approximation space $AS = \langle U, I, \nu \rangle$ is given.

1. $\mathcal{O}_{AS} = \langle U, \mathcal{C}, \sqsubseteq \rangle$, where $\mathcal{C} = \{\langle S_{AS^+}, S_{AS^\oplus} \rangle \mid S \subseteq U\}$ is the set of all approximate concepts over AS, is an approximate ontology.
2. $\mathcal{O}_{AS} = \langle U, \mathcal{C}, \sqsubseteq \rangle$, where

$$\mathcal{C} = \{\langle S_{AS^+}, S_{AS^\oplus} \rangle \mid S \subseteq U \text{ and } Th(S_{AS^+}, S_{AS^\oplus}) \text{ holds}\} \cup \{\textsc{Bottom}, \textsc{Top}\}$$

   is the set of all approximate concepts over AS satisfying a given consistent first-order theory $Th(S_{AS^+}, S_{AS^\oplus})$, is an approximate ontology.
3. $\mathcal{O}_{AS} = \left\langle U, 2^U, \sqsubseteq \right\rangle$, where 2^U is the set of all crisp concepts over U, is a crisp ontology. ▪

Remark 4.1. Background knowledge, such as that expressed in the form of logical theories, is often useful as a means of restricting ontologies, e.g., for filtering out noisy or uninteresting data or concepts.

For example, in a particular application one may not be interested in "small and heavy" objects, and would require that every concept S satisfies the logical constraint[4]

$$\forall x \in S_{AS^+}[small_{AS^+}(x) \to \neg heavy_{AS^\oplus}(x)].$$

If one would like to exclude concepts where both "young" and "old" persons appear, concepts might be required to satisfy the formula

$$\neg\{\exists x \in S_{AS^\oplus} \exists y \in S_{AS^\oplus}[old_{AS^+}(x) \wedge young_{AS^+}(y)]\}.$$

In practical applications, one usually creates ontologies based on particular types of background knowledge or constraints related to the domain of interest. There are many uses of such knowledge such as:

[4]Observe that "small", "heavy", "old" and "young" are concepts as well, thus in the logical language we refer to their lower and upper approximations with the usual intuitive meaning. E.g., $small_{AS^+}(x)$ states that it is certain that x is $small$ and $small_{AS^\oplus}(x)$ states that it might be the case that that x is $small$.

- providing criteria for accepting or removing certain objects from data sets used in building ontologies (e.g., for preprocessing, filtering out noisy data, etc.)
- logical constraints represented as particular theories which concepts should satisfy. Examples were shown above
- providing a definition of similarity between objects that can be used to define concepts approximately as we will see shortly
- providing one with measures of the usefulness of concepts, which can then be used as a filter on sets of concepts, retaining only those that satisfy the filter's conditions. ∎

Definition 4.2. Let $AS = \langle U, I, \nu \rangle$ be an approximation space. For any approximate ontology $\mathcal{O}_{AS} = \langle U, \mathcal{C}, \sqsubseteq \rangle$ and an approximate set E over AS,

- by the *most specific approximate concept in* $\mathcal{O}_{AS}$ *including* E, denoted by $\text{MSC}_{\mathcal{O}_{AS}}(E)$, we understand the smallest wrt $\sqsubseteq$ approximate concept including E, i.e., $E \sqsubseteq \text{MSC}_{\mathcal{O}_{AS}}(E)$ and for any approximate concept E' such that $E \sqsubseteq E'$ we have that $\text{MSC}_{\mathcal{O}_{AS}}(E) \sqsubseteq E'$
- by the *most general approximate concept in* $\mathcal{O}_{AS}$ *included in* E, denoted by $\text{MGC}_{\mathcal{O}_{AS}}(E)$, we understand the greatest wrt $\sqsubseteq$ approximate concept included in E, i.e., $\text{MGC}_{\mathcal{O}_{AS}}(E) \sqsubseteq E$ and for any approximate concept E' such that $E' \sqsubseteq E$ we have that $E' \sqsubseteq \text{MSC}_{\mathcal{O}_{AS}}(E)$. ∎

Frequently we are interested in ontologies, where for any two concepts the respective lower and upper approximations of those concepts are disjoint, or one concept is a generalization/specialization of the other concept.

Definition 4.3. Let $AS = \langle U, I, \nu \rangle$ be an approximation space. By a *strict approximate ontology* over AS we understand an approximate ontology $\mathcal{O}_{AS} = \langle U, \mathcal{C}, \sqsubseteq \rangle$ such that for any $C, D \in \mathcal{C}$ the following condition holds:

$$C_{AS^+} \cap D_{AS^\oplus} \neq \emptyset \text{ implies } C \sqsubseteq D \text{ or } D \sqsubseteq C \tag{1}$$

∎

Strict ontologies provide us with a simple and intuitive method for classifying objects as belonging to particular concepts. Assume that $\mathcal{O}_{AS} = \langle U, \mathcal{C}, \sqsubseteq \rangle$ is a strict approximate ontology. In order to find an approximate concept to which an object $e \in U$ belongs, it is sufficient to find the most specific concept $\langle S, T \rangle \in \mathcal{C}$ such that $e \in S$. If such a concept exists, e can be classified as belonging to $\langle S, T \rangle$ as well as to any generalization of $\langle S, T \rangle$.

Given an approximate ontology $\mathcal{O}_{AS}$, one can construct a "canonical" strict ontology which reflects the contents of the original ontology as closely as possible. Intuitively, one can consider an ontology to consist of agents, where each agent is responsible for delivering information about the contents of a single concept in the ontology. Given $e \in U$ each agent answers whether e belongs to the lower or upper approximations of the concept it is responsible for. If there is no violation of (1), then e is uniquely classified to a concept, otherwise it can be classified to the upper approximation of concepts serviced by "conflicting" agents.

The following algorithm generates the strict *canonical ontology determined by* a given approximate ontology.

Algorithm 4.1.

– Input:
  - an approximation space $AS = \langle U, I, \nu \rangle$
  - an ontology $\mathcal{O}_{AS} = \langle U, \mathcal{C}, \sqsubseteq \rangle$

– Output: the canonical approximate ontology $\mathcal{O}'_{AS} = \langle U, \mathcal{C}', \sqsubseteq \rangle$ determined by $\mathcal{O}_{AS}$

– Algorithm:
  1. $\mathcal{C}' := \mathcal{C}$
  2. for every $e \in U$ do
     begin
       $\mathcal{S} := \{\langle D_{AS^+}, D_{AS^\oplus} \rangle \in \mathcal{C} \mid e \in D_{AS^\oplus}\}$;
       if there are $\langle D_{AS^+}, D_{AS^\oplus} \rangle, \left\langle D'_{AS^+}, D'_{AS^\oplus} \right\rangle \in \mathcal{S}$ such that
         $e \in D_{AS^+}$
         and $\langle D_{AS^+}, D_{AS^\oplus} \rangle \not\sqsubseteq \left\langle D'_{AS^+}, D'_{AS^\oplus} \right\rangle$
         and $\left\langle D'_{AS^+}, D'_{AS^\oplus} \right\rangle \not\sqsubseteq \langle D_{AS^+}, D_{AS^\oplus} \rangle$
       then $\mathcal{C}' := (\mathcal{C}' - \mathcal{S}) \cup \left\{ \textsc{Mgc}_{\mathcal{O}'_{AS}}(\langle D_{AS^+} - \{e\}, D_{AS^\oplus} \rangle) \mid \langle D_{AS^+}, D_{AS^\oplus} \rangle \in \mathcal{S} \right\}$
     end. ∎

This algorithm can be implemented to run in time $O\big(\mid U \mid * \mid \mathcal{C} \mid \big)$.

Observe that in the last line of the algorithm, tuples of the form $\textsc{Mgc}_{\mathcal{O}'_{AS}}(\langle D_{AS^+} - \{e\}, D_{AS^\oplus} \rangle)$ are added to $\mathcal{C}'$. This follows from the fact that approximate concepts are approximate sets, i.e., have to be obtained by means of approximation operations, as defined in Definitions 2.1 and 2.3.

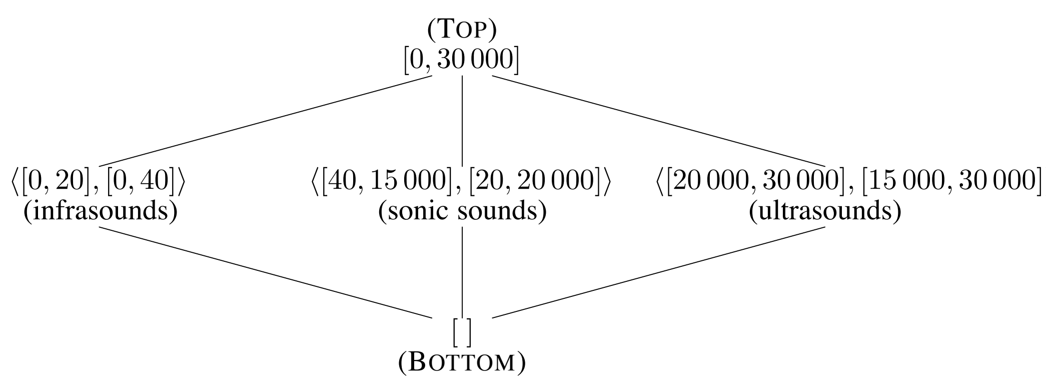

Figure 1. Approximate ontology considered in Example 4.2.

Example 4.2. Consider the audio frequency domain. It is a continuous domain, but hearing boundaries give rise to three discrete categories: infra-sounds (frequency below $20\,\text{Hz}$), sonic sounds (frequency within the interval $[20\,\text{Hz}, 20\,000\,\text{Hz}]$) and ultra sounds (frequency over $20\,000\,\text{Hz}$). It is also known that it is more likely for adults that the frequency of sonic sounds is in the interval $[40\,\text{Hz}, 15\,000\,\text{Hz}]$.

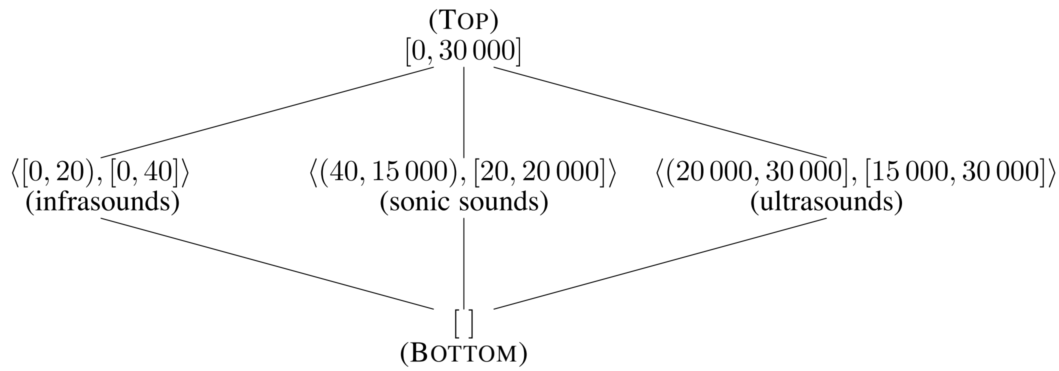

Figure 2. Strict approximate ontology obtained from ontology shown in Figure 1.

Assume $AS = \langle U, I\rangle$ is a standard approximation space, where U consists of integers in the interval $[0, 30\,000]$ and for any $u \in U$, $I(u) = \{u\}$. It is then the case that on the basis of the available data, one could generate an ontology like the one shown in Figure 1, which is not strict. Algorithm 4.1 transforms this ontology into its strict counterpart as shown in Figure 2. ∎

5. Generating Approximate Ontologies

A number of problems arise when one attempts to generate approximate ontologies for a given domain with a given data set as input. The following problems are two of the more pressing:

- the number of approximate concepts generated may be large (exponential in the size of the domain of objects)
- in general, many of the approximate concepts generated are artificial, as they may contain totally unrelated objects.

There are a number of ways to alleviate these problems. In the following, we show two methods for generating approximate ontologies:

1. the first method (see Section 5.1) transforms a given crisp ontology into a strict approximate ontology.
2. the second method (see Section 5.2) generates approximate ontologies only using raw data as the initial input.

5.1. Generating Approximate Ontologies from Set-Based Ontologies

Definition 5.1. By a *set-based* ontology we understand any tuple of the form $\langle U, \mathcal{C}, \sqsubseteq\rangle$, where $\mathcal{C} \subseteq 2^U$ is a set of subsets of U such that $\emptyset, U \in \mathcal{C}$, and $\sqsubseteq$ is the classical set-theoretical inclusion. ∎

Let $\langle U, \mathcal{C}, \subseteq \rangle$ be a *set-based ontology*. These are the most common types of ontologies and many can be found and downloaded from the Internet. Suppose an agent (human or software) has additional knowledge about similarities between objects in the base domain for the ontology. In addition, assume that this information is provided by an approximation space $AS = \langle U, I, \nu \rangle$. Such similarity constraints are very useful in many application domains. For instance, in robotics domains, certain limitations may be present in the types of sensors and cameras used. For example, the colors red and orange may be indiscernible. If other concepts in a given ontology are dependent on this concept distinction, they will make no sense to the robot and its reasoning capabilities will be improperly modeled.

One way to deal with this problem is to provide an approximation space which represents the indiscernability of red and orange and use this to transform a set-based ontology into a (strict) approximate ontology. Namely, it suffices to consider the approximate ontology, where classical sets are replaced respectively by their approximations, i.e., the approximate ontology

$$\mathcal{O}_{AS} = \langle U, \{\langle S_{AS^+}, S_{AS^\oplus} \rangle \mid S \in \mathcal{C}\}, \sqsubseteq \rangle .$$

One can then turn $\mathcal{O}_{AS}$ into its corresponding strict approximate ontology simply by using Algorithm 4.1.

5.2. Generating Approximate Ontologies from Data

In section 5.1, we started with a great deal of structure, a set-based ontology, before generating an approximation to it. Quite often, one begins with much less structure, raw data sets for instance, and some additional knowledge about similarity between the objects in the raw data set. As before, we will represent this information as an approximation space and use it to generate an approximate ontology.

Let $AS = \langle U, I, \nu \rangle$ be an approximation space. The process of generating an approximate ontology using only a data set and AS as input is based on the following ideas:

- in the first step, base concepts have to be generated. There may be many ways to do this, but we assume some method suitable for the application domain and type of data is chosen.[5] For the purpose of this paper, we will use a relatively straightforward means for generating base concepts. Given AS, we consider the set, $\{I(u) \mid u \in U\}$ as including all base concepts. These are concepts representing neighborhood sets for each individual based on some similarity notion;

- the base concepts generated in the previous step will serve as a basis for generating more general concepts. Namely, in the next level of the hierarchy we consider $I'(u')$, for $u' \in 2^U$, i.e., a concept of concepts similar to the concept u' and so on. This idea can be applied recursively generating additional levels in the hierarchy of concepts.

In order to do this construction recursively, power approximation spaces are introduced.

Definition 5.2. Let $AS = \langle U, I, \nu \rangle$ be an approximation space. By the *power approximation space over* AS we understand approximation space $2^{AS} \stackrel{\text{def}}{=} \left\langle 2^U, I', \nu' \right\rangle$, where:

[5] Among them very intuitive would be to generate base concepts as clusters (see [11]) or as cliques in a "similarity graph" $G = \langle U, E \rangle$, where $E \stackrel{\text{def}}{=} \{\langle u, u' \rangle \in U \mid u' \in I(u) \text{ or } u \in I(u')\}$. However, generating cliques is, in general, not tractable.

- $I'(S) \stackrel{\text{def}}{=} \{T \in 2^U \mid \nu(S,T) > 0\}$
- $\nu'(S,T) \stackrel{\text{def}}{=} \begin{cases} \dfrac{\mid \{s \in S \mid \exists t \in T[s \in I(t)]\} \mid}{\mid S \mid} & \text{when } S \neq \emptyset \\ 1 & \text{when } S = \emptyset. \end{cases}$ ∎

We also require the following definition of a flat representation of any finite family of sets of higher-order types.

Definition 5.3. Let U be a finite set, let $P_0(U) \stackrel{\text{def}}{=} U$ and let for any natural number n, $P_{n+1}(U) \stackrel{\text{def}}{=} 2^{P_n(U)}$. A *flat representation* of a finite set $X \in P_n(U)$, denoted by $flat(X)$, is defined inductively:

$$flat(X) \stackrel{\text{def}}{=} \begin{cases} \{u\} & \text{for } X = u \in P_0(U) \\ \bigcup\limits_{Y \in X} flat(Y) & \text{for } X \in P_n \text{ with } n > 1. \end{cases}$$

∎

For example, $flat\Big(\Big\{ \{\{4\}\}, \{\{3\}, \{4,5\}\} \Big\}\Big) = \{3,4,5\}$.

The following algorithm will generate an approximate ontology from an approximation space, where the domain of individuals in the approximation space is viewed as the initial raw data set.

Algorithm 5.1.

– Input: an approximation space $AS = \langle U, I, \nu \rangle$
– Output: a set-based ontology $\mathcal{O} = \langle U, \mathcal{C}, \subseteq \rangle$
– Algorithm:
  1. $\mathcal{C} := \{I(u) \mid u \in U\}$;
     $i := 0; AS_i := AS; \mathcal{C}_i := \mathcal{C}$;
  2. while new concepts are added to $\mathcal{C}$ do
     begin
       $AS_{i+1} := 2^{AS_i}; \mathcal{C}_{i+1} := \emptyset$;
       for all $u \in \mathcal{C}_i$ do
       begin
         $\mathcal{C} := \mathcal{C} \cup flat(I_{i+1}(u))$;
         $\mathcal{C}_{i+1} := \mathcal{C}_{i+1} \cup I_{i+1}(u)$
       end;
       $i := i + 1$;
     end;
  3. $\mathcal{C} := \mathcal{C} \cup \{\emptyset, U\}$.

∎

Now, using the method described in Section 5.1, transform the set-based ontology $\mathcal{O}$ obtained using algorithm 5.1 into the (strict) corresponding approximate ontology $\mathcal{O}_{AS}$.

Remark 5.1. It is not yet known whether Algorithm 5.1 is tractable.[6] In the examples we have considered so far, the algorithm has a polynomial time complexity. A similar situation applies to the use of tolerance spaces used in section 8.1, to generate approximate ontologies. However, in that case we are able to show some sufficient conditions which guarantee tractability. ∎

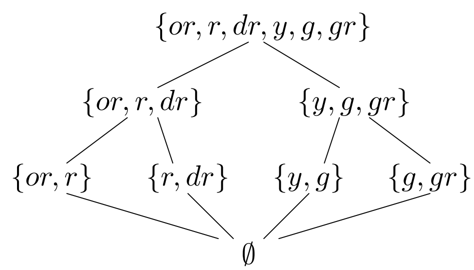

Figure 3. Set-based ontology obtained in Example 5.1.

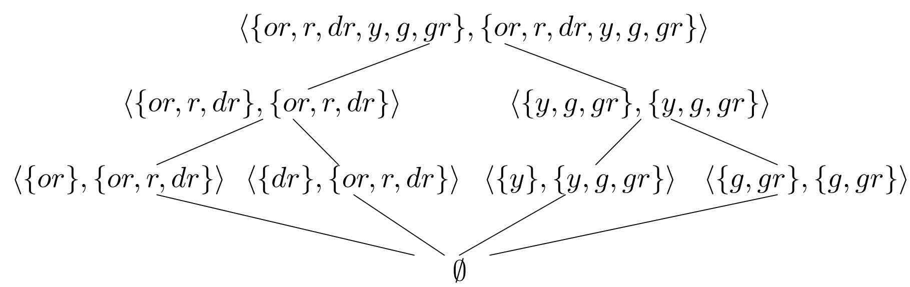

Figure 4. Approximate ontology obtained from set-based ontology shown in Figure 3.

Example 5.1. Let AS be the approximation space considered in Example 3.1. After the first step of Algorithm 5.1, $\mathcal{C} = \big\{\{or, r\}, \{or, r, dr\}, \{r, dr\}, \{y, g\}, \{g, gr\}\big\}$. In the while loop (second step) we first consider the power approximation space $AS_1 = \langle 2^U, I_1\rangle$, where:

$$I_1(\{or, r\}) = I_1(\{or, r, dr\}) = I_1(\{r, dr\}) = \big\{\{or, r\}, \{or, r, dr\}, \{r, dr\}\big\}$$
$$I_1(\{y, g\}) = I_1(\{g, gr\}) = \big\{\{y, g\}, \{g, gr\}\big\}\ .$$

Of course, $flat\big(\{\{or, r\}, \{or, r, dr\}, \{r, dr\}\}\big) = \{or, r, dr\}$ and $flat\big(\{\{y, g\}, \{g, gr\}\}\big) = \{y, g, gr\}$. Thus, after the first iteration, $\mathcal{C} = \big\{\{or, r\}, \{or, r, dr\}, \{r, dr\}, \{y, g\}, \{g, gr\}, \{y, g, gr\}\big\}$. The second iteration does not add anything new, so, after the third step, the set-based ontology which is output contains concepts $\mathcal{C} = \big\{\emptyset, \{or, r\}, \{or, r, dr\}, \{r, dr\}, \{y, g\}, \{g, gr\}, \{y, g, gr\}, \{or, r, dr, y, g, gr\}\big\}$. Figure 3 shows this ontology and Figures 4, 5 show the corresponding approximate ontology and strict approximate ontology. ∎

[6] Of course, statement $AS_{i+1} := 2^{AS_i}$ appearing in Algorithm 5.1 does not require to represent explicitly the whole power approximation space 2^{AS_i}.

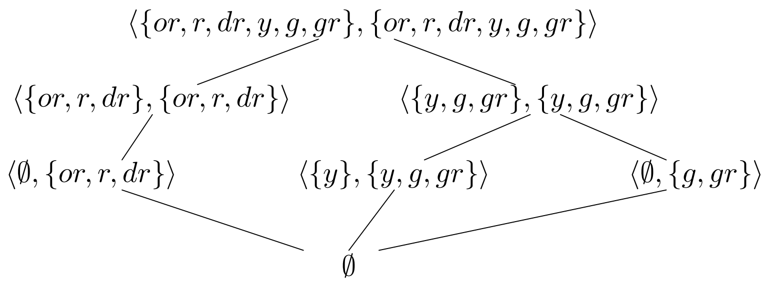

Figure 5. Strict approximate ontology obtained from set-based ontology shown in Figure 4.

6. Tolerance Spaces

There is an important and very useful specialization of approximate ontologies where approximations are based on a definition of similarity induced by the use of tolerance spaces.

The following definitions were provided in [3].

Definition 6.1. By a *tolerance function* on a set U we mean any function $\tau : U \times U \longrightarrow [0,1]$ such that for all $x, y \in U$, $\tau(x,x) = 1$ and $\tau(x,y) = \tau(y,x)$. ▪

Definition 6.2. A *tolerance space* is defined as the tuple $TS = \langle U, \tau, p \rangle$, which consists of

- a nonempty set U, called the *domain* of TS;
- a *tolerance function* τ
- a *tolerance threshold* $p \in [0,1]$. ▪

Tolerance spaces are used to construct tolerance neighborhoods for individuals.

Definition 6.3. Let $TS = \langle U, \tau, p \rangle$ be a tolerance space. By a *neighborhood function wrt* TS we mean a function given by

$$n_{TS}(u) \stackrel{\text{def}}{=} \{u' \in U \mid \tau(u,u') \geq p \text{ holds}\}.$$

By a *neighborhood of* u *wrt* TS we mean the value $n_{TS}(u)$. ▪

Observe that any tolerance space can be considered an approximation space. One of many possible definitions follows.

Definition 6.4. Let $TS = \langle U, \tau, p \rangle$ be a tolerance space. By an *approximation space induced by* TS we understand the standard approximation space $AS_{TS} = \langle U, I_{AS} \rangle$, where for any $u \in U$, $I_{AS}(u) \stackrel{\text{def}}{=} n_{TS}(u)$. The lower and upper approximations of a set S wrt TS, S_{TS^+} and $S_{TS^\oplus}$, are defined according to Definition 2.2, i.e.,

$$S_{TS^+} = \{u \in U : n_{TS}(u) \subseteq S\}$$
$$S_{TS^\oplus} = \{u \in U : n_{TS}(u) \cap S \neq \emptyset\}.$$ ▪

Observe also that any pair of the form $\langle S_{TS^+}, S_{TS^\oplus}\rangle$ is an approximate set in the sense of Definition 2.3.
In the cognitive science literature (see, e.g., [8]), some types of categorical perception are hypothesized to occur when

> "there is a quantitative discontinuity in discriminability at the category boundaries of a physical continuum, as measured by a peak in discriminative acuity at the transition region for the identification of members of adjacent categories."

Perception and categorization can then often be seen as a transformation of continuous quantitative structure into a discrete qualitative one. The following example shows how tolerance spaces can contribute to model such situations as well as illustrate the use of tolerance spaces and approximations.

7. Tolerance-Based Ontologies

Based on Definition 6.4, tolerance concepts and tolerance-based ontologies can be defined as approximation concepts and approximate ontologies over an approximation space induced by a given tolerance space. However, explicit definitions are provided below since tolerance spaces offer such a useful representation of similarity.

Definition 7.1. Let $TS = \langle U, \tau, p\rangle$ be a tolerance space. By a *tolerance concept* over TS we understand any pair $\langle S_{TS^+}, S_{TS^\oplus}\rangle$, where $S \subseteq U$. ▪

It is easily observed that tolerance concepts are approximate concepts.

Definition 7.2. Let $TS = \langle U, \tau, p\rangle$ be a tolerance space. By a *tolerance-based ontology* wrt TS, denoted by $\mathcal{O}_{TS}$, we understand this to be any approximate ontology, where approximate concepts are required to be tolerance concepts. ▪

8. Generating Tolerance-Based Ontologies

8.1. Generating Tolerance-Based Ontologies from Data

In this section we provide a method for generating ontologies from raw data, polynomial in the size of the domain, and which is also based on tolerance spaces.

In order to construct tolerance-based ontologies, we require a definition of power tolerance spaces which will be provided below. It should be emphasized that there is no "canonical" definition of this notion, suitable for all application domains. The definitions given below are chosen because of their simplicity and the fact that their properties allow for the tractable construction of tolerance-based ontologies.

Definition 8.1. Let $TS = \langle U, \tau, p\rangle$ be a tolerance space and let $U_1, U_2 \subseteq U$. By the *inclusion function induced by* TS we mean the function given by

$$\nu_{TS}(U_1, U_2) \stackrel{\text{def}}{=} \begin{cases} \dfrac{|\{u_1 \in U_1 \mid \exists u_2 \in U_2[u_1 \in n_{TS}(u_2)]\}|}{|U_1|} & \text{if } U_1 \neq \emptyset \\ 1 & \text{otherwise.} \end{cases}$$ ▪

Definition 8.2. Let $TS = \langle U, \tau, p \rangle$ be a tolerance space. By a *power tolerance space induced by* TS we mean $2^{TS} = \left\langle 2^U, \tau, s \right\rangle$, where

- for $U_1, U_2 \in 2^U$, $\tau(U_1, U_2) \stackrel{\text{def}}{=} \min\{\nu_{TS}(U_1, U_2),\ \nu_{TS}(U_2, U_1)\}$
- $s \in [0, 1]$ is a tolerance parameter. ■

Compared to the method described in Section 5, tolerance spaces are easier to tune due to the explicit tolerance threshold allowing one to identify or discriminate given (sets of) objects.

If neighborhoods $I(u)$ used in Algorithm 5.1 are defined to be $n_{TS}(u)$ then tractability is not guaranteed (see Remark 5.1). However we have the following sufficient condition which guarantees tractability:

> if the $I(u)$ used in Algorithm 5.1 are defined to be the minimal sets X wrt inclusion such that $u \in X$ and $X = X_{TS^+}$, where TS is a power tolerance space of a suitable order, then the time complexity of the resulting algorithm is polynomial wrt the size of the underlying domain of objects.[7]

8.2. Generating Tolerance-Based Ontologies from Information Systems

Let us now assume that the set of objects which will be used to generate ontologies is given by means of an information system defined as follows.

Definition 8.3. An *information system* is any pair $\mathcal{A} = \langle U, A \rangle$, where U is a non-empty finite set of *objects*, called the *universe*, and A is a non-empty finite set of functions, called *attributes*, such that $a : U \to U_a$ for every $a \in A$. The set U_a is called the *value set* of a. ■

Information systems are often represented as tables with the first column containing objects and the remaining columns, separated by vertical double lines, containing values of attributes. Such tables are called *information tables*.

In this section, we will assume that tolerance spaces on objects appearing in information systems are provided. One method of obtaining such tolerance spaces would be to provide tolerance spaces on each attribute domain and combine them into a single tolerance space.

Below we will illustrate this approach, using the following definition.[8] Note that the definition of tolerance on tuples is heavily application-dependent. We use this particular definition in order to instantiate the approach and show that even such a simple definition results in interesting ontologies.

Definition 8.4. Let $\mathcal{A} = \langle U, A \rangle$ be an information system and let $TS_{a_1} = \langle U_{a_1}, \tau_{a_1}, p_{a_1} \rangle, \ldots, TS_{a_k} = \langle U_{a_k}, \tau_{a_k}, p_{a_k} \rangle$ be tolerance spaces for all attributes $a_1, \ldots, a_k \in A$. By a *tolerance space on* $\mathcal{A}$ *induced by* $TS_{a_1}, \ldots, TS_{a_k}$ we mean the tolerance space

$$TS^{\mathcal{A}}_{TS_{a_1} \ldots TS_{a_k}} = \langle U_{a_1} \times \ldots \times U_{a_k}, \tau, q \rangle,$$

where

[7] Observe that in the case of tolerance spaces, if $X = X_{TS^+}$ and $Y = Y_{TS^+}$ and $X \neq Y$ then $X \cap Y = \emptyset$. This assures that in each iteration of the algorithm the number of concepts added, decreases.

[8] Based on the definition of tolerance spaces on tuples, given in [3].

object	income	history	employment	security
o_1	4	5	5	4
o_2	2	2	1	2
o_3	5	4	5	4
o_4	2	3	3	4
o_5	2	3	2	3

Table 1. Information table containing row data for Example 8.1.

- $\tau(\langle u_1, \ldots, u_k\rangle, \langle u'_1, \ldots, u'_k\rangle) \stackrel{\text{def}}{=} \begin{cases} \dfrac{|\{u_i : 1 \leq i \leq k \text{ and } u_i \in n_{TS_i}(u'_i)\}|}{k} & \text{if } k \neq 0 \\ 1 & \text{otherwise.} \end{cases}$
- $q \in [0, 1]$ is a tolerance parameter. ■

Remark 8.1. In the cognitive science literature, (see, e.g., [8, 10]) one often distinguishes between two different kinds of categories (concepts):

- "all-or-none" categories, where members share a common set of features (attributes) and a corresponding rule defines these as necessary and sufficient conditions for membership
- "graded" categories, where membership is a matter of a similarity degree.

Observe that the approach based on tolerance spaces allows us to deal with both cases uniformly. Namely, assume a certain set of features (attributes of an information system) is not to be considered as a basis for membership in an "all-or-none" category. One simply needs to define a tolerance space for each of these attribute's value sets where all values in the respective sets are similar to each other. In this manner, the attributes in question become irrelevant in the discernibility process. ■

Example 8.1. Consider the information system $\mathcal{A}$ given in Table 1, containing information about bank clients requiring a new loan. All attributes are evaluated on the scale $1, \ldots, 5$, with 1 indicating the worst and 5 the best grade. Let

$$TS \stackrel{\text{def}}{=} TS_{income} = TS_{history} = TS_{employemnt} = TS_{security} = \langle\{1, 2, 3, 4, 5\}, \tau_1, 0.7\rangle$$

be the same tolerance space for all attributes, where for $x, y \in \{1, \ldots, 5\}$,

$$\tau_1(x, y) \stackrel{\text{def}}{=} 1 - \frac{|\ x - y\ |}{4}.$$

Consider the tolerance space on $\mathcal{A}$ induced by TS,[9]

$$TS^{\mathcal{A}}_{TS} = \left\langle \{1, 2, 3, 4, 5\}^4, \tau, q \right\rangle,$$

[9]In fact, the tolerance space in question is induced by $TS_{income}, TS_{history}, TS_{employemnt}, TS_{security}$, but we simplify the notation here.

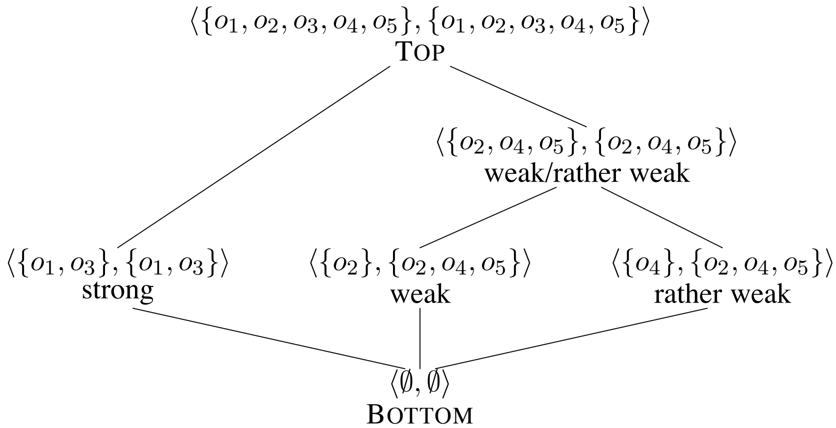

Figure 6. Approximate ontology considered in Example 8.1.

where τ is defined as in Definition 8.4 and we take $q = 0.75$. Then:[10]

$$n(o_1) = n(o_3) = \{o_1, o_3\},\ n(o_2) = \{o_2, o_5\},\ n(o_4) = \{o_4, o_5\},\ n(o_5) = \{o_2, o_4, o_5\}.$$

Assuming tolerance thresholds in all power tolerance spaces equal to 1.0, one obtains the following set of concepts of the generated set-based ontology:

$$\left\{\emptyset, \{o_1, o_3\},\ \{o_2, o_5\},\ \{o_4, o_5\},\ \{o_2, o_4, o_5\},\ \{o_1, o_2, o_3, o_4, o_5\}\right\},$$

which gives rise to the following approximate ontology (see also Figure 6, where we also provide intuitive labels, such as "weak", "rather weak", "strong", which explain the results):

$$\begin{aligned}&\left\{\ \langle\emptyset, \emptyset\rangle, \langle\{o_1, o_3\}, \{o_1, o_3\}\rangle, \langle\{o_2\}, \{o_2, o_4, o_5\}\rangle, \langle\{o_4\}, \{o_2, o_4, o_5\}\rangle,\right.\\ &\left.\langle\{o_2, o_4, o_5\}, \{o_2, o_4, o_5\}\rangle, \langle\{o_1, o_2, o_3, o_4, o_5\}, \{o_1, o_2, o_3, o_4, o_5\}\rangle\right\}.\end{aligned}$$

Observe that the above ontology is strict. ■

9. Conclusions

In this paper, we presented a formal framework for defining approximate concepts, ontologies and operations on approximate concepts and ontologies. The framework is based on intuitions from rough set theory, but generalizes the notion of indiscernability used in rough set theory by using approximation spaces and a specialization of them called tolerance spaces. Algorithms for automatically generating approximate ontologies from traditional crisp ontologies or from large data sets together with additional knowledge was also presented. The knowledge was generally related to similarity measurements between individual objects in the data sets and is represented by approximation or tolerance spaces, in

[10]Below $n(x)$ abbreviates $n_{TS^{\mathcal{A}}_{TS}}(x)$.

addition to constraints of a logical nature which rule out particular constellations of concepts and dependencies in generated ontologies. Approximate concepts and ontologies provide a rich generalization of their crisp counterparts and are appropriate for many applications, including perception, knowledge structuring, practical reasoning, classifier construction and robotics. The general techniques presented here for generating approximate ontologies on the basis of raw data are parameterizable. We are convinced that there are many additional instantiations of the techniques specific to different application domains. Consequently, additional experimentation and empirical evaluation of the approach presented is desirable and this is work that will be pursued in the future.

References

[1] Berners-Lee, T.: *Weaving the Web*, HarperBusiness, 2000.

[2] Catteano, G.: Abstract Approximation Spaces for Rough Theories, *[15]*, 1998.

[3] Doherty, P., Łukaszewicz, W., Szałas, A.: Tolerance spaces and approximative representational structures, *Proceedings 26th German Conference on Artificial Intelligence* (A. Günter, R. Kruse, B. Neumann, Eds.), 2821, Springer-Verlag, 2003.

[4] Duentsch, I., Gediga, G.: Uncertainty Measures of Rough Set Prediction, *Artificial Intelligence*, **106**, 1998, 77–107.

[5] Fensel, D.: *Ontologies: A Silver Bullet for Knowledge Management and Electronic Commerce*, Springer-Verlag, 2001.

[6] Gruber, T. R.: A Translation Approach to Portable Ontology Specifications, *Knowledge Acquisition*, **5**, 1993, 199–220.

[7] Harnad, S., Ed.: *Categorical Perception: The Grundwork of Cognition*, Cambridge University Press, 1987.

[8] Harnad, S.: Psychophysical and cognitive aspects of categorical perception: A critical overview, *[7]*, 1987.

[9] Lutz, C., Wolter, F., Zakharyaschev, M.: A tableau algorithm for reasoning about concepts and similarity, *Proceedings of the 12th International Conference on Automated Reasoning with Analytic Tableaux and Related Methods TABLEAUX 2003*, LNAI, Springer Verlag, 2003.

[10] Medin, D., Barsalou, L.: Categorization Processes in Category Structure, *[7]*, 1987.

[11] Mitchell, T.: *Machine Learning*, Mc Graw-Hill, Portland, 1997.

[12] Pal, S., Peters, J., Polkowski, L., Skowron, A.: Rough-Neural Computing: An Introduction, *[13]*, 2003.

[13] Pal, S., Polkowski, L., Skowron, A., Eds.: *Rough-Neuro Computing: Techniques for Computing with Words*, Cognitive Technologies, Springer–Verlag, 2003.

[14] Pawlak, Z.: *Rough Sets. Theoretical Aspects of Reasoning about Data*, Kluwer Academic Publishers, Dordrecht, 1991.

[15] Polkowski, L., Skowron, A., Eds.: *Rough Sets in Knowledge Discovery 1: Methodology and Applications*, vol. 17 of *Studies in Fuzziness and Soft Computing*, Physica-Verlag, Heidelberg, 1998.

[16] Skowron, A., Stepaniuk, J.: Generalized approximation spaces, *Soft Computing: Rough Sets, Fuzzy Logic, Neural Networks, Uncertainty Management, Knowledge Discovery* (T. Lin, A. Wildberger, Eds.), 1995.

[17] Skowron, A., Stepaniuk, J.: Tolerance Approximation Spaces, *Fundamenta Informaticae*, **27**, 1996, 245–253.